Instruktion för instuderingsmaterial

Fakta

Kurs: Systemutvecklare C/C++ Extended 2024
Klass: SUVx24
Teknikområde: Versionshantering i Git

Learning Target

Versionshantering i Git

Hur fungerar det att använda versionshantering med systemet Git?

Introduktion

Att samarbeta kring kod kan vara utmanande om man ska utveckla den tillsammans i ett team. Det har funnits flera olika sätt att hantera detta historiskt. Olika verktyg har tagits fram för att hjälpa till med den praktiska hanteringen av att göra versioner av källkoden och en som blivit väldigt populär på senare tid är Git. Den används gärna tillsammans med GitHub för att ha en gemensam plattform att ytterligare främja arbetsprocessen med källkodens versionshantering.

Material

Här nedan hittar du ditt självstudiematerial. Du ska studera materialet i den ordning det listas för att du på bästa sätt ska tillgodogöra dig materialet.

Instruktion för instuderingsmaterial

Learning Target

Vad innebär versionshantering? Hur använder vi ***Git*** och ***GitHub*** tillsammans med ***VS Code***?

Hur

Du behöver inte göra något praktiskt under instuderingen för det ska vi göra på Workshop 13. Du behöver bra läsa och titta på materialet nedan och vara påläst, men inte kunna allt utantill.

Teoretiskt

Studera vilka vanliga begrepp och namn som används i samband med versionshantering genom att använda länkarna nedan (både text och video) och svara på tillhörande frågor.

Googla på sökfrasen: code versioning

Vilka termer hittar du för att benämna hur vi hanterar versioner av kod?

Praktiskt

För att det ska bli så lätt som möjligt för er att följa med i exempel och övningar framöver, vänta gärna till nästa Worskhop med att installera ***Git*** på era datorer. Då undviker vi skillnader som stör i form av inställningar och onödig felsökning.

Vi kommer på Workshop 13 att

- installera ***Git*** och göra gemensamma val under installationen
- skapar ett individuellt ***GitHub*** konto där det gäller att välja en lämlig epostadress
- integrera ***Git*** med ***VS Code*** med tillhörande plugin.

Om vi gör dessa saker tillsammans kommer övningar bli lättare att följa och utbildaren kan lättare hjälpa till vid problem.

Instruktion för instuderingsmaterial

Versionshantering

Text

What is version control? Definition, types, systems and tools - LogRocket Blog

Läs artikeln och svara på:

- Vad är versionshantering?
- Av vilka anledningar vill vi hålla på med versionshantering av kod?
- Vilka olika huvudformer av versionshantering finns det?
    - Fördelar och nackdelar?
- Läs om fördelar att använda versionshantering i projektstyrning.
- Vilka är det populäraste versionshanteringssystemen?
    - Spelar det någon roll vilken vi väljer?
- Finns det något man måste beakta när man ska börja använda versionshantering?

Git, GitHub och versionshantering

Text

What is Git? A Beginner's Guide to Git Version Control

- Vad är ***Git*** respective ***GitHub***?
    - Vad är deras användningsområde?
- Vilka sätt kan vi interagera med ***Git*** på?
    - Är det ena sättet bättre än det andra?
    - Vilket sätt föredrar du?
- Hur förbereder man för anvädning av ***Git***?
- Vad är ***Git*** *repository*?
- Hur kan man samarbeta med andra kring *repository*?
- Beskriv för dig själv det typiska arbetsflödet i ***Git***.
- Vad innebär (*Commit*) *History* i ***Git***?
- Vad är syftet med *branches*?

Instruktion för instuderingsmaterial

Video

Git and GitHub Tutorial for Beginners (ca 45 min)

Versionshantering

- Vad är ***SCM***?
- Vad är ***Git Bash***?
- Vilka olika vektyg kan du använda ***Git*** tillsammans med?

Git

- Vilka inställningar är bra att göra i ***Git Bash***?
- Hur kan du kolla olika inställningar i konfigurationen?
- Vilka olika ***Git***-kommandon visas i videon?
- Vad gör `.gitignore`?
- Vad innebär *staging*?
- Till vad används en *branch*?
- Vad innebär en *merge*?
- Hur uppstår en *merge conflict*?
    - Hur kan man lösa den?

GitHub

- Vad kan du använda ***GitHub*** till ?
- Hur får du koden från din dator till ett *repository* på ***GitHub***?
- Vad är ett *remote repository*?
- På vilka sätt kan koden flyttas med mellan *local* och *remote repository*?
- Vad ä en *pull request*?
    - Hur kopplar *issue* till *pull request*?
- Vad mer kan du gör på ***GitHub***?
- Hur får du ner kod från ett *repository* ***GitHub*** till din dators *repository*?

Mer om Git och Github

Om du vill få ungefär samma information, fast med en annan beskrivning, om versionshantering, Git och GitHub kan du även titta på följande videoklipp:

GitHub Basics Made Easy: A Fast Beginner's Tutorial! (ca 24 min)

OBS! Använder ***GitHub Desktop*** i demonstrationer i stället för ***VS Code*** som vi kommer att använda.

- Vad innebär versionshantering?
- Vilka kommandon är nödvändiga för att kunna utföra versionshantering med Git som *solo developer*?
- Hur inträffar en *merge conflict*?
- Vad är *branching*?
- Var ska du undvika att spara ett klonat projekt från ***Github***?
    - Varför?
- Vad är proceduren för att kolla om det finns något nytt i ett *repository* och sedan ladda ner det nya?
    - Kan man alltid ladda upp ny/ändrad kod till ett klonat projekt?

Git Tutorial For Dummies (ca 20 min)

- Vilka olika ***Git***-kommandon används i videon?
- Vad används respektive kommando till?
- Vad är det för skillnad på att använda ***Git*** och ***GitHub***?
- Vad är ett *repository*?
- Vad innebär att skapa och använda en *branch*?
- Vad gör kommandot *merge*?